M1Secil

UNIVERSITÉ TOULOUSE III PAUL SABATIER Université de Toulouse

Examen Synthèse des systèmes matériels

Session 1 / Jeudi 16 mars 2023
15:45 - 17:45 bât.U4 amphi BAILLAUD

Documents non autorisés

I - Cours

Les questions sont indépendantes les unes des autres et peuvent donc être traitées dans le désordre.

a) On considère la portion de code ci-contre :

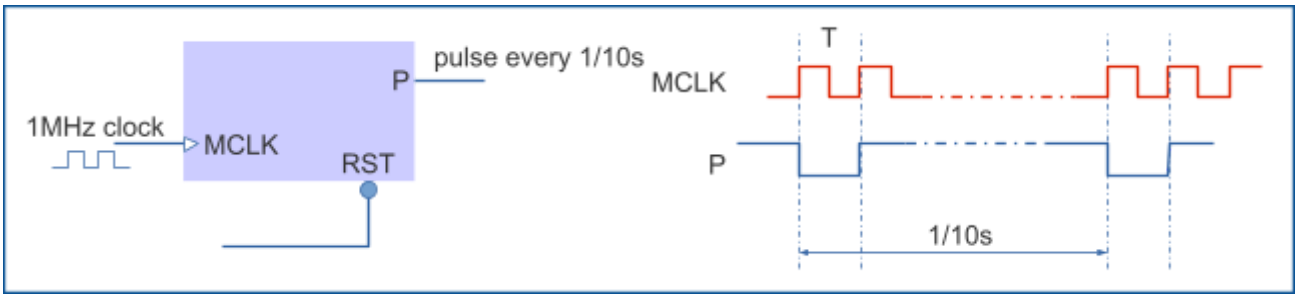
```
QA <= (others=> 'X') when is_x(ADR_A) else
      (others=> '0') when conv_integer(ADR_A)=0 else
      REGS(conv_integer(ADR_A)) when D2QA=FALSE else
      D;
```

Réécrivez cette affectation conditionnelle de la sortie QA dans le domaine séquentiel (i.e un *process*).

b) En quoi une sortie dite '**à collecteur ouvert**' diffère-t-elle d'une sortie dite '***push-pull***' (ou *totem-pole*) ? Étayez vos propos d'un schéma et indiquez pour chacune d'elle tous les états possibles du signal en sortie.

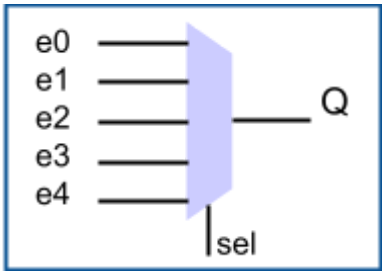
c) Dans la description d'une entité d'un composant se trouve, notamment, la définition des ports d'entrées / sorties. Expliquez la différence entre un port en mode **`inout`** par rapport à un port en mode **`buffer`**. Étayez vos propos d'un schéma tel que vu en cours.

d) Décrivez, en VHDL, l'entité et l'architecture du composant présenté ci-dessous:



e) On considère la portion de code suivante:

```
type SELECTOR is (e0, e1, e2, e3, e4);
signal sel:SELECTOR;
```



Quelle est, en **nombre de bits**, la taille du signal `sel` ?
Décrivez en VHDL, dans le domaine concurrent, le multiplexeur présenté ci-contre.

f) Expliquez en quoi consiste l'opération dite de "synthèse logique" ?

z) Suite à une synthèse, le '*timing report*' fait apparaître la notion de **slack**: de quoi s'agit-il ? mieux vaut-il qu'il soit positif ou négatif ?? (*pas de réponse à pile ou face svp!)*

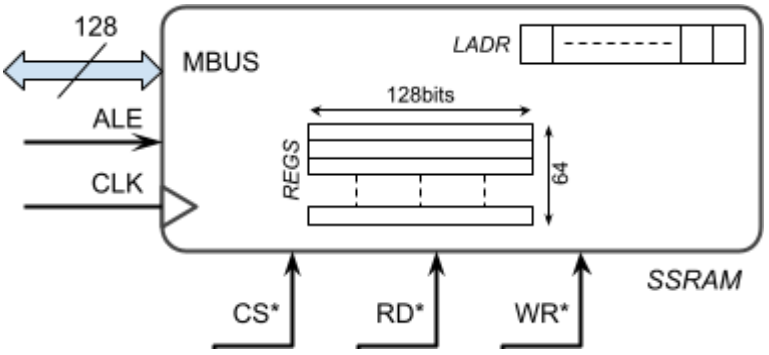
tournez la page svp .. /..

II - Multiplexed Synchronous Static RAM with burst transfers

On considère une mémoire statique synchrone faisant usage **d'un seul bus** pour les données et les adresses (i.e bus multiplexé). Les caractéristiques génériques de ce composant sont les suivantes :

- **MBUS_WIDTH** Taille du bus multiplexé `[MBUS]` avec par défaut **128 bits**,
- **RAM_SIZE** Nombre de mots mémoire (de taille MBUS) avec par défaut **64 mots**,

Principe de fonctionnement



Hormis pour le **reset**, toute opération est conditionnée par la présence du signal `CS*` actif, soit `CS*='0'`.

`[MBUS]` étant multiplexé, l'**adresse** est lue sur les **bits de poids faible** de `[MBUS]` tant que le signal `ALE` est actif. Les données pourront ensuite être lues ou écrites via `[MBUS]` dès l'activation du signal `RD*` ou `WR*`. Une fois un cycle lecture ou écriture amorcé, **l'adresse est tacitement incrémentée** (i.e *burst*) jusqu'à la taille maximale de la SSRAM.

Un cycle **reset** est amorcé par la présence des deux signaux `RD*` et `WR*` actifs simultanément au moment du front montant de l'horloge `CLK`: `RD*=WR*='0'` sur front montant `CLK` □ **reset**

> **Notes**
>
> Les flèches doubles définissent un signal bidirectionnel *inout*.
> '*' associé à un signal dénote que ce dernier est **actif à l'état bas**.

Implémentation

`CS*='1'` SSRAM non activée: `[MBUS] <= 'Z'` & aucune opération n'est prise en compte hormis **reset**.
`CS='0' RD=WR='1'` le CPU positionne l'adresse de départ sur `[MBUS]` et active `ALE`.
Lors d'une écriture (`WR*='0'`), les données sont échantillonnées par la SSRAM sur front montant d'horloge.
Les lectures (`RD*='0'`) sont combinatoires: le CPU récupèrera les données présentées sur `[MBUS]` sur front montant de `CLK`. Les signaux `ALE`,`CS*`, `RD*` et `WR*` sont pilotés par le CPU et changent d'état peu après le front montant de `CLK`.

1) Compte-tenu des valeurs par défaut des paramètres génériques, quelle est, en nombre d'octets, la taille de cette SSRAM ?

2) Quelle est, en VHDL, la signature du signal `LADR` qui va échantillonner l'adresse sur `[MBUS]` ?

3) Vous coderez en VHDL l'entité et l'architecture de ce composant en veillant à minimiser les ressources utilisées (i.e attention aux entiers non bornés !).

```
entity muxed_ssram is
    generic(
        MBUS_WIDTH: _____,
        RAM_SIZE: _______ );
    port(
        MBUS: _____,
        _____ );
end muxed_ssram;

architecture behaviour of muxed_ssram is

    _____
begin
_______
_______
end behaviour;
```